Mentor Graphics (Holdings) Ltd. et al.
Our Ref.: G 1094 PCT

System and Method of Clocking an IP Core During a Debugging Operation

The present invention relates to a system and a method of clocking an IP core during a debugging operation.

In the design of integrated circuits, there is an increasing demand for emulation and verification tools. Hardware-based verification solutions have been around for years in two different embodiments: accelerators and emulators. Useful tools in emulation systems are so-called IP-Xpress kits which are emulation-ready kits for concurrent hardware and software verification of processor-based systems. Such IP-Xpress kits use microprocessor or DSP chips, mounted onto a printed-circuit-board to provide the functionality of the device to be connected to a design mapped into the emulator, and the kits consist of a board and HDL-wrapper files.

All processor-type IP cores require some external clock source. Hence, one or more clock signals are provided to the IP-Xpress boards. The clocks are provided either directly from the internal clock generators of the emulator or they may be driven from the design loaded onto the emulator.

Another way of providing clock to an IP core is by using a clock oscillator mounted onto the IP-Xpress board. In this case any frequency can be applied, i.e. there are no maximum clock frequency constraints due to the emulation system.

A key advantage of an IP-Xpress kit is to provide a fast running system verification environment in which application software is running on the IP core and this stimulating the design mapped to the emulator. In case of faulty system behaviour, the cause for this could either be in the application software or in the design (provided the IP core is functioning correctly). In order to identify the erroneous component of the system both the application software as well as the design has to be debugged. This can be done most conveniently when the hardware and software are stopped synchronously. On the one hand, this gives a good correlation between the design's status and the actual software execution, and on the other hand it enables to interrogate all resources of the design mapped onto the emulator system. This may mean, however, that the emulator clocks are stopped. In case the IP core is clocked by a clock generated by the emulator system, this would mean that the IP core is not clocked anymore. Hence, the software debugger would not necessarily work and as a result the resources of the software execution were not visible and a full system debugging would not be possible.

It is the object of the present invention to provide a system and a method of clocking an IP core during a debugging operation. This object is achieved with the features of the claims.

In order to achieve this object, according to the present invention, the clock generation for the IP core when entering the system debugging mode is implemented on the IP-Xpress board itself. There is provided a clock oscillator and a switching means, and the switching means switches to the clock oscillator provided on the IP-Xpress board as soon as the system debug mode is entered. In order to initiate the switching operation, the switching means monitors signals specific to the IP core which indicate a breakpoint, e.g., EMU0/1 in case of TI C6x DSPs, and if these indicate that a breakpoint has been entered, the clock output of the switching means is driven from the clock oscillator. Hence, the IP core is continuously clocked even when the clocks of the emulator system are stopped. Furthermore, the software debugger is still functional in its system debug mode and all IP core internal resources and the software execution status can be investigated.

Upon leaving the system debug mode, the switching means is signalled to switch back to the clocks of the emulator system, and the system execution can continue in its normal operational mode.

The present invention will now be described with reference to the figure. The figure shows a preferred embodiment of the present invention comprising a switching means 1. Connected to the switching means is a clock oscillator 2. On the left of the figure there are shown two possible clock sources for the normal operation of the emulator system. These clock signals are driven through the backplane of the emulator system onto the IP-Xpress board. The signal gpc2i_clkin is a driven clock from the design, and gpc2i_j17_clkin is a clock directly from the generator of the emulator. Depending on the select signal supplied to the multiplexer 3 either of these clocks is provided to the IP core. For example, the IP core is a DSP. The clock oscillator 2, the switching means 1 and the signals indicating a breakpoint form the clock generator when entering the breakpoint mode. The output of the switching means is a clock signal that clocks the IP core.

During normal operation of the emulator system, the switching means feeds a “regular” clock through its output. When the system debug mode is entered, the switching means switches from this “regular” clock to the clock oscillator provided on the IP-Xpress board.

The following code fractions demonstrate how the switching to the clock oscillator can be implemented inside the switching means.

```
-- -----------------------------------------------------
-- Memorize breakpoint ( ie: !EMU0 or !EMU1)
-- -----------------------------------------------------
process(RESET, EMU_LATCH_RST, OSC_CLK)

begin

  if (RESET = '0' or EMU_LATCH_RST = '0') then
    emu_trigger <= '0';
  elsif rising_edge(OSC_CLK) then
    if (EMU0 = '0' or EMU1 = '0' or emu_trigger = '1') then
      emu_trigger <= '1';
    else
      emu_trigger <= '0';
    end if;
  end if;

end process;

-- ----------------------------------------------------
-- Latch emu_trigger with the falling edge of CLKIN1
-- ----------------------------------------------------
process(RESET,CLKIN1)

begin

  if (RESET = '0') then
    clk_sel1 <= '0';
  elsif falling_edge(CLKIN1) then
    clk_sel1 <= emu_trigger;
  end if;

end process;

-- ----------------------------------------------------
-- Select the right clock for int_clk_for dsp1.
-- When the emulator is running select CLKIN1
-- else, when a breakpoint occurs, select OSC_CLK
-- so that the connection with the software debugger
-- is not lost
-- ----------------------------------------------------
process(CLKIN1, OSC_CLK, clk_sel1)

begin

  case clk_sel1 is
    when '0' =>
      int_clk_for_dsp1 <= CLKIN1;
    when '1' =>
      int_clk_for_dsp1 <= OSC_CLK;
    when others =>
      int_clk_for_dsp1 <= '0';
  end case;

end process;
```